Zero Configuration Networking (Rendezvous)

Session 811

Zero Configuration Networking (Rendezvous)

Stuart Cheshire
Apple Networking Engineer
Co-chairman, IETF Zeroconf

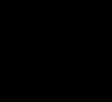
Introduction

- Why?
  - Local Area Communications (0–10m) chaos
- What?
  - Wide Area has converged on IP as the One True Protocol—why not Local Area too?
- How?
  - How do we make IP meet this challenge?
- APIs

Why?

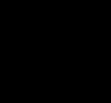
Wide Area Convergence

- DECNET
- Xerox XNS
- TCP/IP
- OSI
- ATM-to-the-desktop
- IBM/Microsoft NetBEUI
- AppleTalk
- Etc., etc., etc.



Wide Area Convergence

- DECNET
- Xerox XNS
- TCP/IP
- OSI
- ATM-to-the-desktop
- IBM/Microsoft NetBEUI
- AppleTalk
- Etc., etc., etc.



Local Area Chaos

- Serial Ports
- Parallel Ports
- SCSI
- ADB

Local Area Chaos

- Serial Ports
- Parallel Ports
- SCSI
- ADB

Local Area Chaos

- USB
- FireWire
- Ethernet
- IrDA
- AirPort (aka “WiFi”, “802.11”, etc.)

Local Area Chaos

- USB
- FireWire
- Ethernet
- IrDA
- AirPort (aka “WiFi”, “802.11”, etc.)
- Bluetooth

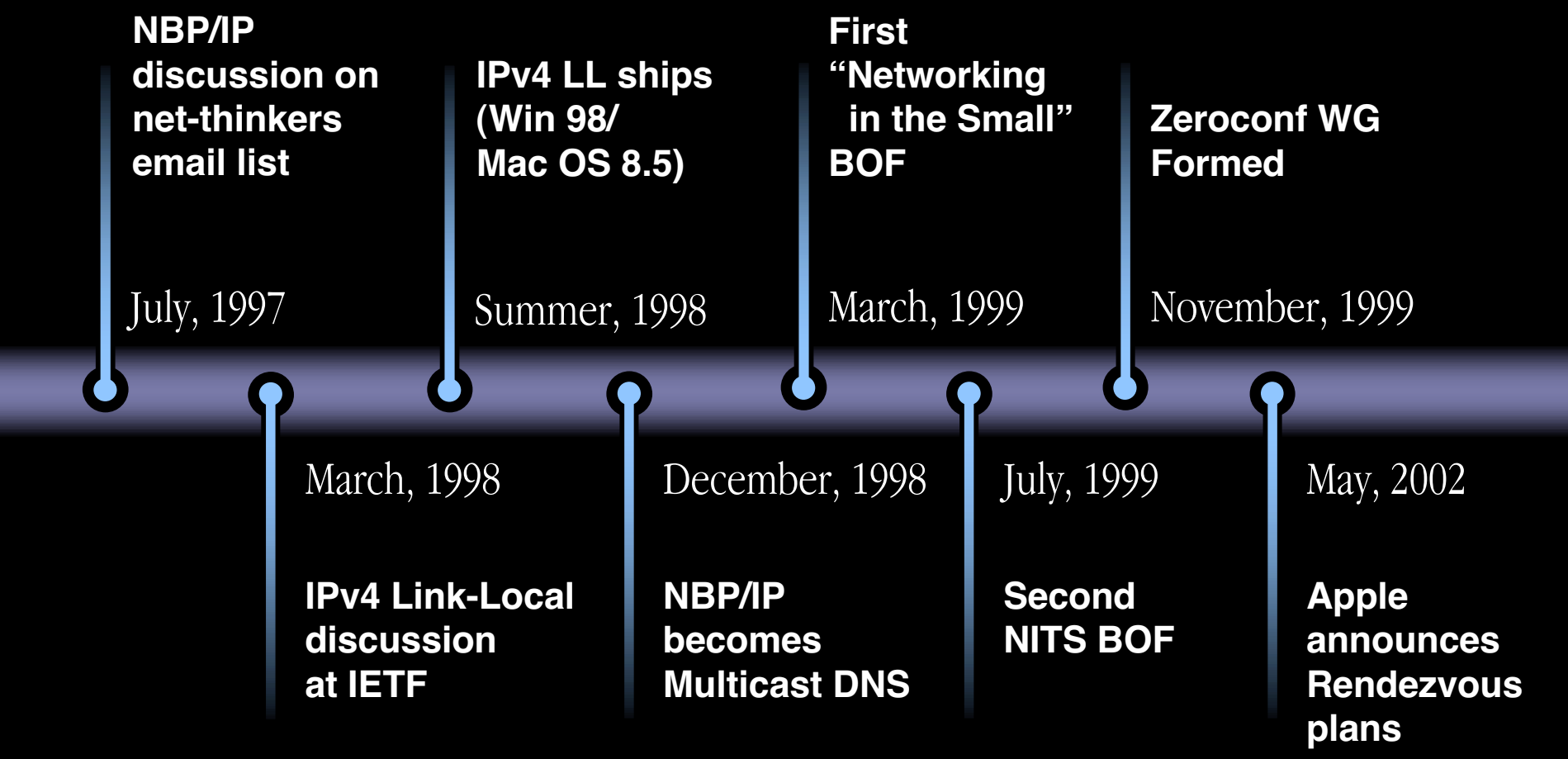
What?

Pick One Protocol

- Wide Area Communications have already converged on IP
- Why not Local Area Communications too?

How?

The Schedule



Three Legs of Zeroconf

- Requirements
  - Addressing
  - Naming
  - Browsing
- draft-ietf-zeroconf-reqts-10.txt
- **http://www.zeroconf.org/**

Addressing

- Self-Assigned Link-Local Addressing
  - Pick random address in 169.254/16
  - ARP to see if anyone else is using it
  - If someone else is using it, try again
  - Ongoing conflict checking
- draft-ietf-zeroconf-ipv4-linklocal-05.txt
- **http://www.zeroconf.org/**

IPv4 Link-Local Availability

- Self-Assigned IPv4 Link-Local Addressing first appeared in:
  - Mac OS 8.5, Summer 1998
  - Windows 98, Summer 1998
  - Mac OS X 10.0
  - Ongoing development for Linux, etc.
- IPv6 already has Link-Local Addressing

Naming

- Multicast DNS
  - Pick desired name in ".local.arpa." subdomain
  - Issue Query to see if anyone else is using it
  - If someone else already using it, pick another
  - Ongoing conflict checking
- draft-cheshire-dnsext-multicastdns-00.txt
- **http://www.multicastdns.org/**

Multicast DNS Availability

- Multicast DNS client
  - On your Mac OS X Jaguar Developer CDs
  - Reportedly planned for Windows XP update
- Just type “laserwriter.local.arpa.” into your Web browser

Demo

Browsing

- Raising the bar
- Should not need to know name in advance
- This is not a new idea . . .

Browsing

- Long list of attempts
  - RLP (Resource Location Protocol)
  - RDP (Resource Discovery Protocol)
  - SLP (Service Location Protocol)
  - SDP (Simple Discovery Protocol)
  - SSDP (Simple Service Discovery Protocol)
  - SDS (Service Discovery Service)

Browsing

- Long list of attempts
  - HAVi
  - Jini
  - e-speak
  - Salutation
  - UPnP
  - OSGi

Why Did They Fail?

- Failed to keep things simple
  - Couldn't resist urge to keep adding features
  - A 450×450 testing matrix is 100,000 tests!
- Failed to understand basic principles
  - Browse for services, not for devices
  - Browsing to find services and using a service are two different operations
  - They could have learned this by just copying AppleTalk Name Binding Protocol (NBP)

DNS Service Discovery

- Already agreed that devices need to implement mDNS
- Can we use that same code for Service Discovery?

DNS Service Discovery

- Use standard DNS capabilities
  - Multiple Answers
  - PTR Records
  - SRV Records (RFC 2782)
    - Escape the tyranny of Well Known Ports

Browsing via DNS PTR

- DNS Query:

  _printer._tcp.local.arpa. PTR ?

Browsing via DNS PTR

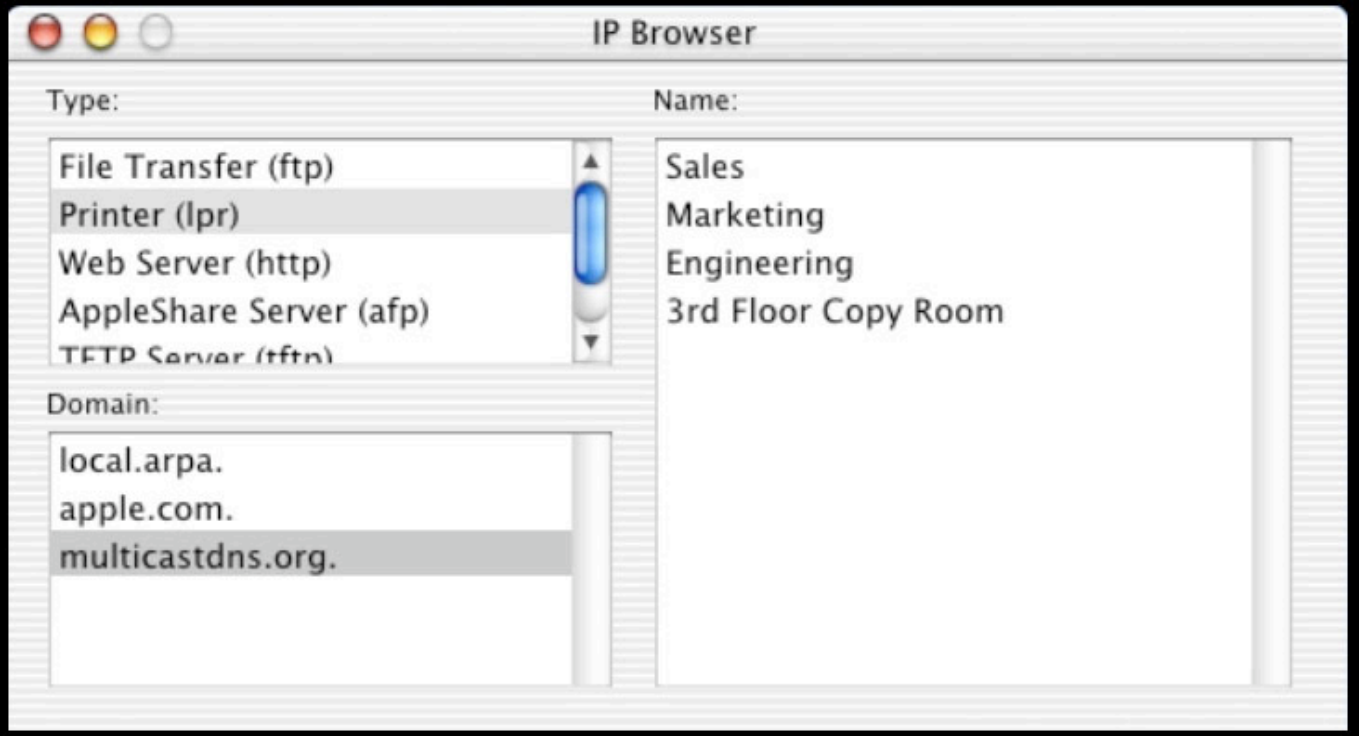
- DNS Response(s):

_printer._tcp.local.arpa. PTR
Sales._printer._tcp.local.arpa.
Marketing._printer._tcp.local.arpa.
Engineering._printer._tcp.local.arpa.
3rd Floor Copy Room._printer._tcp.local.arpa.

Browsing via DNS PTR

- DNS Response(s):

_printer._tcp.local.arpa. PTR
Sales._printer._tcp.local.arpa.
Marketing._printer._tcp.local.arpa.
Engineering._printer._tcp.local.arpa.
3rd Floor Copy Room._printer._tcp.local.arpa.


IP Browser
Type:
File Transfer (ftp)
Printer (lpr)
Web Server (http)
AppleShare Server (afp)
Domain:
local.arpa.
apple.com.
multicastdns.org.
Name:
Sales
Marketing
Engineering
3rd Floor Copy Room

Components of Service Name

- User-Visible Name
  - First label of name—Unicode (UTF-8) text
  - Equivalent to NBP Name
- Service Type/Service Protocol Name
  - Second two labels of name
  - Equivalent to NBP Type
- Domain
  - Remainder of name
  - Equivalent to NBP Zone

Components of Service Name

- User-Visible Name

  3rd Floor Copy Room._printer._tcp.local.arpa.

- Service Type/Service Protocol Name

  3rd Floor Copy Room._printer._tcp.local.arpa.

- Domain

  3rd Floor Copy Room._printer._tcp.local.arpa.

Service Types

- A Service is identified by what protocol it uses
- Protocol names are assigned by IANA
  - Internet Assigned Numbers Authority
- List of assigned protocol names

  http://www.iana.org/assignments/port-numbers
- Applying for your own is easy

  http://www.iana.org/cgi-bin/usr-port-number.pl

Avoiding ‘Chattiness’

- Aggressive Caching
- Duplicate Suppression Section in Queries
- Duplicate Answer Suppression
- Duplicate Question Suppression
- Exponential query backoff
- Service announcement on startup
- Exponential announcement backoff

Lookup via DNS SRV

- DNS Queries

  Sales._printer._tcp.local.arpa. SRV ?
  Sales._printer._tcp.local.arpa. TXT ?

Lookup via DNS SRV

- DNS Responses

  Sales._printer._tcp.local.arpa. SRV
  0 0 515 host.local.arpa.

  Sales._printer._tcp.local.arpa. TXT lpt1

  host.local.arpa. A 169.254.12.34

Demo

DNS-SD Availability

- System-provided API for Jaguar
- Any client or service that wants to implement it:
  - draft-cheshire-dnsext-nias-00.txt
  - Darwin Open Source in Jaguar time-frame
  - mDNS and DNS-SD, client and server:
    ≈ 3000 lines of C
    ≈ 50K compiled PowerPC
  - Hardware vendors: Contact Apple!
- **http://www.dns-sd.org/**

Debugging Multicast DNS

- Standard DNS Packet Format
- On UDP Port 5353 (instead of 53)
- View the packets using EtherPeek
- Edit EtherPeek.app/Contents/decoders/IETF.dcd

```
TEQU  19    2  0  0  SMTP;
TEQU  35    2  0  0  DNS;
TEQU  14e9  2  0  0  DNS;
TEQU  50    2  0  0  HTTP;
TEQU  6d    2  0  0  POP;
```

- **http://www.multicastdns.org/**

Zeroconf for Headless Devices

- AirPort Base Station, cable modem, firewall . . .
- Ethernet printer, network video Camera . . .
- Rack-mount server
    - No more Serial Port Terminal Connections!
- Networked TiVos
- . . . and many more

Demo

Stuart Cheshire and Erik Peyton

APIs

Low-level Mach Message API

```
dns_service_discovery_ref DNSServiceFoo(...);

port = DNSServiceDiscoveryMachPort(ref);

void MyHandleMachMessage(port, ... )
    {
    DNSServiceDiscovery_handleReply(msg);
    }

DNSServiceDiscoveryDeallocate(ref);
```

Registering a Service

```
dns_service_discovery_ref
  DNSServiceRegistrationCreate
(
    char                    *name,
    char                    *regtype,
    char                    *domain,
    Opaque16                 port,
    char                    *txtRecord,
    DNSServiceRegReply       callBack,
    void                    *context
);
```

Browsing

```
dns_service_discovery_ref
  DNSServiceBrowserCreate
(

    char                        *regtype,
    char                        *domain,


    DNSServiceBrowserReply   callBack,
    void                        *context
);
```

Resolving

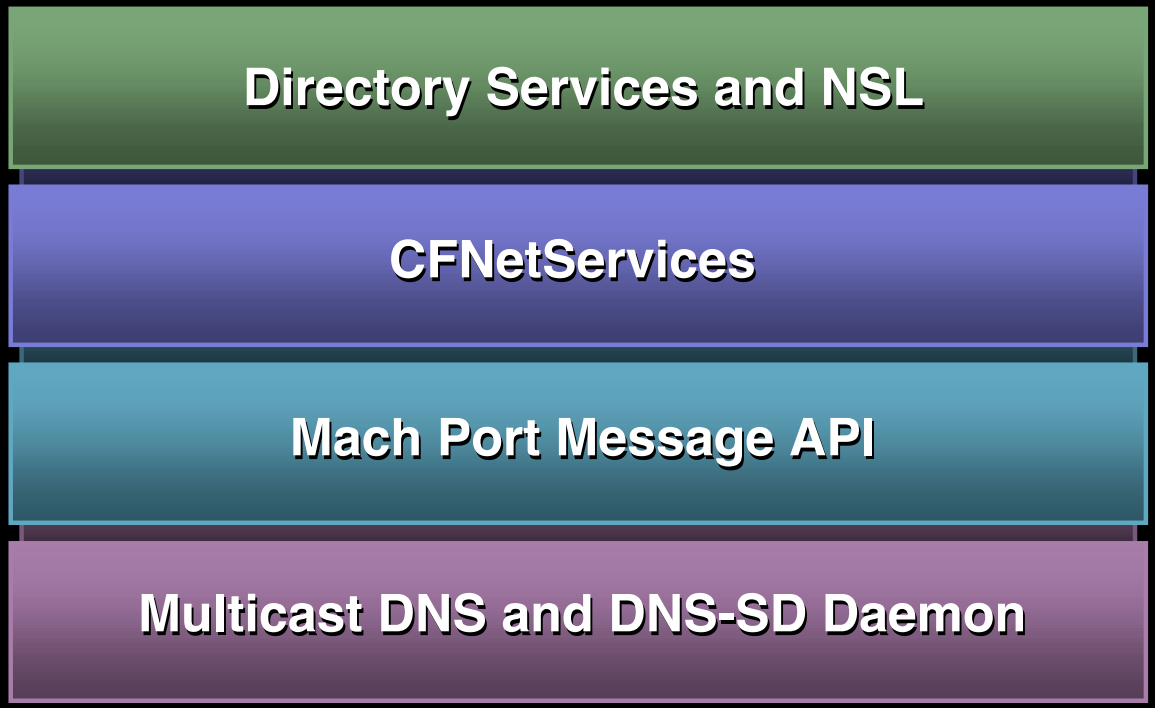
```
dns_service_discovery_ref
  DNSServiceResolverResolve
(
    char                            *name,
    char                            *regtype,
    char                            *domain,


    DNSServiceResolverReply  callBack,
    void                            *context
);
```

Enumerating Domains

```
dns_service_discovery_ref
  DNSServiceDomainEnumerationCreate
(
    int
   registrationDomains,
    DNSServiceDomEnumReply callBack,
    void                                *context
);
```

DNS Service Discovery

API Layers in Jaguar



CFNetServices

Jeremy Wyld
CFNetwork Engineer

What Is CFNetwork?

- Abstractions for network protocols
- APIs in the style of CoreFoundation
- Part of CoreServices.framework
- Used by higher level frameworks
    - NSL, Directory Services, NSNetServices

What Is CFNetServices?

- Abstraction on top of the DNSDiscovery API
- Two types of objects
  - CFNetService
    - Instance of a service entity
  - CFNetServiceBrowser
    - Instance of a search for services or domains

Using CFNetServices

- Basic usage
  - Create
  - Schedule
  - Run
  - Handle callbacks
- Three main tasks
  - Advertise your service
  - Find an existing service
  - Connect to a service you found

Advertising Your Service

```
void  registerMyService(CFStringRef name, UInt32 port,
      void* info) {

      CFStreamError error;
      CFNetServiceContext ctxt = {0, info, NULL, NULL, NULL};

      CFNetServiceRef myService =
            CFNetServiceCreate(kCFAllocatorDefault,
                  CFSTR(“”), CFSTR(“_http._tcp.”), name, port);

      CFNetServiceSetClient(myService, &registerCB, &ctxt);

      CFNetServiceScheduleWithRunLoop(myService,
            CFRunLoopGetCurrent(),
                  kCFRunLoopCommonModes);

      CFNetServiceRegister(myService, &error);
}
```

Advertising Your Service

```
void registerServiceCB(CFNetServiceRef service,
        CFStreamError* error, void* info) {

        if ((error->domain == kCFStreamErrorDomainNetServices)
        &&
                (error->error == kCFNetServicesErrorCollision))
        {
                // Formulate a new name.
                ...
                registerMyService(newName, gPort, info);
        }

                CFNetServiceSetClient(service, NULL, NULL);

                CFNetServiceUnscheduleFromRunLoop(service,
                        CFRunLoopGetCurrent(),
                kCFRunLoopCommonModes);

                CFRelease(service);
}
```

Finding an Existing Service

```
void findServices(void* info) {

    CFStreamError error;
    CFMutableArrayRef list = CFArrayCreateMutable
        (kCFAllocatorDefault, 0, &kCFTypeArrayCallBacks);
    CFNetServiceContext ctxt = {0, list, CFRetain,
        CFRelease, CFCopyDescription};

    CFNetServiceBrowser myBrowser =
        CFNetServiceBrowserCreate(kCFAllocatorDefault,
            &myBrowserCB, &ctxt);

    CFNetServiceBrowserScheduleWithRunLoop
        (myBrowser, CFRunLoopGetCurrent(),
            kCFRunLoopCommonModes);

    CFNetServiceBrowserSearchForServices(myBrowser,
        CFSTR(“”), CFSTR(“_http._tcp.”), &error);
}
```

Finding an Existing Service

```
void myBrowserCB(CFNetServiceBrowserRef browser,
        CFOptionFlags flags, CFTypeRef service,
        CFStreamError* error, void* info) {

        CFMutableArrayRef list = (CFMutableArrayRef)info;

        if (flags & kCFNetServiceFlagRemove) {
                // Remove the service from the browser list
                CFArrayRemoveValueAtIndex(list, service);

        } else {
                // Add the service to the browser list
                CFArrayAppendValue(list, service);
        }

        if ((flags & kCFNetServiceMoreComing) == 0)
                // Update the UI to reflect the changed services
}
```

Connecting to a Service

```
void  resolveAService(CFStringRef name, void* info) {

      CFStreamError error;
      CFNetServiceContext ctxt =
            {0, info, NULL, NULL, NULL};

      CFNetServiceRef theService =
            CFNetServiceCreate(kCFAllocatorDefault,
                  CFSTR(“”), CFSTR(“_http._tcp.”), name, 0);

      CFNetServiceSetClient(theService, &resolveCB, &ctxt);

      CFNetServiceScheduleWithRunLoop(theService,
            CFRunLoopGetCurrent(),
            kCFRunLoopCommonModes);

      CFNetServiceResolve(theService, &error);
}
```

Connecting to a Service

```
void resolveCB(CFNetServiceRef service, CFStreamError* error, void*
info) {

    if (error->error == noErr) {
        CFSocketContext ctxt = {0, info, NULL, NULL, NULL};

        CFSocketRef s = CFSocketCreate(kCFAllocatorDefault,
        PF_INET, SOCK_STREAM, IPPROTO_TCP,
        kCFSocketConnectCallBack | kCFSocketReadCallBack |
        kCFSocketWriteCallBack, &socketCB, &ctxt);

        CFArrayRef allAddrs = CFNetServiceGetAddressing(service);
        CFDataRef addr = CFArrayGetValueAtIndex(allAddrs, 0);

        CFSocketConnectToAddress(s, addr, -1);
    }
    else {
        …
    }
}
```

Connecting to a Service

```
void resolveCB(CFNetServiceRef service, CFStreamError* error, void*
info) {

    if (error->error == noErr) {
        …
    }
    else {
        CFNetServiceSetClient(service, NULL, NULL);

        CFNetServiceUnscheduleFromRunLoop(service,
        CFRunLoopGetCurrent(), kCFRunLoopCommonModes);

        CFRelease(service);
    }
}
```

The Road Ahead . . .

- Software Developers
  - Use DNS-SD for network browsing in your Mac OS X applications
- Hardware Developers
  - Build all three legs of Zeroconf into your hardware products

Roadmap

803 Mac OS X Networking Overview	Room A2 **Tue., 9:00am**
805 Introducing CFNetwork	Room C **Tue., 5:00pm**
809 Advanced Mac OS X Networking	Room C **Thurs., 9:00am**

Whom to Contact

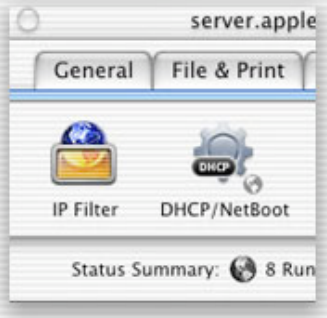
Tom Weyer
Network and Communications Evangelist
weyer@apple.com

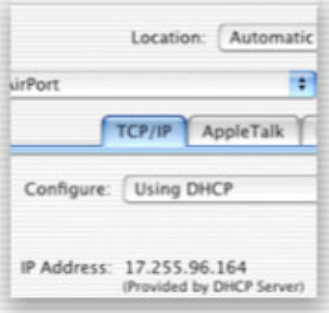
http://developer.apple.com/wwdc2002/urls.html

For More Information

- Jaguar Low-level Mach Message API
  DNSServiceDiscovery.h
- IETF Web Pages
  http://www.ietf.org/
- IANA Registry
  http://www.iana.org/
- Other Places
  - **http://www.zeroconf.org/**
  - **http://www.multicastdns.org/**
  - **http://www.dns-sd.org/**

Q&A





Tom Weyer
Network and Communications Evangelist
weyer@apple.com

http://developer.apple.com/wwdc2002/urls.html


WWDC2002


WWDC2002


WWDC2002